CERT'S PODCASTS: SECURITY FOR BUSINESS LEADERS: SHOW NOTES

Mainstreaming Secure Coding Practices

Key Message: Requiring secure coding practices when building or buying software can dramatically reduce vulnerabilities.

Executive Summary

"Easily avoided software defects are a primary cause of commonly exploited software vulnerabilities. CERT has observed, through an analysis of thousands of vulnerability reports, that most vulnerabilities stem from a relatively small number of common programming errors. By identifying insecure coding practices and developing secure alternatives, software developers can take practical steps to reduce or eliminate vulnerabilities before deployment." [1]

In this podcast, Robert Seacord, the leader of CERT’s Secure Coding Initiative, discusses how to reduce the costs associated with deploying vulnerable software by making secure coding practices part of everyday business.

PART 1: MAKING THE BUSINESS CASE FOR SECURE CODE

Compelling Arguments

Making code more secure is roughly equivalent to improving its quality. Building software better reduces its overall development cost.

Costs incurred by customers include losses resulting from attacks, and installing and managing patches.

Reducing Development Costs

In essence, developing code securely can reduce development costs as evidenced by the SEI’s Team Software Process (TSP). TSP data shows a 78 percent improvement in productivity.

The Software Development Life Cycle: Where to Focus Attention

Understand the security requirements for your system and what information users will be accessing.

With this understanding, you can develop a more secure design which is essential for secure coding practices to be effective.

While useful, source code analysis tools (including static analysis) tend to be relied upon too heavily by software developers. Developers need to understand what the security issues are and how to address them when they are writing the code.

In addition, having a clean bill of health from a static analysis tool does NOT mean that your code is free of defects that could lead to exploited vulnerabilities.

Finding vulnerabilities later in the life cycle forces developers to have to re-familiarize themselves with the code. This takes more time and can result in costly rework and reduced quality.

In other words, it is important to code correctly the first time. This is a tenet of the SEI’s Personal Software Process.

PART 2: WORKING WITH SOFTWARE SUPPLIERS; SELECTING THE RIGHT CODING PRACTICES

Considerations When Buying Software

Make sure to understand and document your requirements. Consider prototyping the system to help better understand these.

Remain engaged during the development process. Be a collaborator with your supplier. Don't disengage.

Compliance as One Avenue

Consider requiring that your supplier conform to a secure coding standard such as the ones CERT has developed for C, C++, and Java.

The State of New York is in the process of requiring its software developers to address the SANS/CWE Top 25 Programming Errors.

In addition, talks are in progress to consider using coding standards as a source to create new regulations.

Top 10 – Top 25 Lists: Upside and Downside

Several organizations have created lists of good software security practices including:

- CWE/SANS Top 25 Most Dangerous Programming Errors
- OWASP (Open Web Application Security Project) Top 10
- CERT Top 10 Secure Coding Practices

Such lists can be useful but are often insufficient. As one example, CERT's C Secure Coding Standard provides 200 guidelines which are prioritized by:

- the consequence of the rule being violated
- the likelihood that a violation can lead to a vulnerability
- remediation costs for correcting the rule violation in existing code

For most organizations, it makes sense to review a complete list of issues and tailor it for their particular application and requirements.

When Is Enough Enough?

Think about how your application will be deployed and assess risks resulting from coding issues that could lead to high value, critical exploits. Mitigate the highest priority risks first.

Forward Progress on Coding Standards

CERT is involved in proposing changes to language standards to make specific programming languages more secure by design. Efforts include:

- INCITS (International Committee for Information Technology Standards): the primary U.S. focus for standardization and interfacing to ISO/IEC standards bodies
  - specifically, PL22.11 (C language) and PL22.16 (C++ language)
- Work on C1X, which is the next major revision of the C standard. The intent is to reflect language changes in C compilers so that once source code is recompiled, it is automatically more secure.

PART 3: STARTING UP A SECURE CODING PROJECT; BARRIERS AND FIRST STEPS

A Big Barrier: Reduced Performance

One of the most significant barriers to implementing secure coding practices is a reduction in software performance.

As one example, Microsoft had several default options in their C compiler which makes the resulting code more secure. They were planning on removing these options because developers were unhappy with the slowdown in performance caused by the options. Organizations need to perform tradeoff analysis to identify solutions that balance adequate performance with adequate security.

Developers have to be willing to accept some performance hit to reduce software vulnerabilities.

Getting Started

The leader of a project to introduce secure coding practices should be the person responsible for software development within the organization.

The first step is to raise developer awareness and offer comprehensive training. Most developers don't understand security risks and how errors they make can create vulnerabilities.

CERT offers a 4-day course in secure coding for C and C++ which covers the basics.

A good second step is to adopt a well-vetted secure coding standard.

Resources

[1] CERT's Secure Coding Initiative

CERT's Secure Coding Standards

Seacord, Robert. *Secure Coding in C and C++*, Addison-Wesley, September 2005.

Seacord, Robert. *The CERT C Secure Coding Standard*. Addison-Wesley, October 2008.

U. S. Department of Homeland Security Build Security In web site